Quest Journals
Journal of Software Engineering and Simulation
Volume 9 ~ Issue 10 (October 2023) pp: 34-40
ISSN(Online) :2321-3795 ISSN (Print):2321-3809
www.questjournals.org

Research Paper

Document Understanding Framework-An Innovative Way to Extract Information from Documents

Himadeep Movva
Independent Researcher

Abstract:
The automation process usually needs input in the form of documents. The information would be extracted from structured, semi-structured, or unstructured documents and ready for a bot to pick up as input data. Using UiPath, we can easily configure rules, including regular expressions, to extract data from digital documents. However, the challenge becomes imminent if the document contains semi-structured, unstructured, or handwritten data. UiPath, with its default features, couldn't extract reliably content in these cases. UiPath's document understanding framework is used to extract data from these documents. Using this framework, the bot can extract information from multiple documents. Instead of creating or configuring different logic for different use cases, this approach is constructive. This paper explores the document understanding framework, usage of ML models for extracting data, and human-in-the-loop validation to act on flagged data, combining Robotic Process Automation (RPA) and Artificial Intelligence (AI) to extract, interpret data from different document types and ensure end to end document processing. Intelligent document processing enables businesses to increase efficiency, reduce errors, and enhance compliance. This research study discusses aspects and essential features of this framework and how they can be a game-changing solution for processing unstructured documents.

Keywords: *Optical Character Recognition, Machine Learning, Document Understanding, Digitization, AI Center, Taxonomy, Training Pipelines, Evaluation Pipelines, Full Pipelines, Machine learning Extractors, ML Skill, and human-in-the-loop*

I. Introduction:

Traditionally, data from documents can be extracted by UiPath Robot using optical character recognition. Some variability in the data, such as changes in the naming of fields across documents, can be configured using regular expressions. However, if there aren't finite variables, it'd be challenging to configure rules for data extraction. Also, in the case of multiple vendors, documents would be received from various vendors, each having a different structure. This would also be a classic example of why traditional approaches to reading data wouldn't work and why a comprehensive framework is required to read data effectively from multiple sources with varying formats. [1]This challenge would be resolved using the UiPath Document Understanding Framework. Even when the documents are unstructured, using in-built OCR and ML models, data can be extracted using predefined or customized models and then used for processing. It has a variety of predefined solutions to extract and classify data from common document types. However, if prebuilt solutions don't work for a specific use case, new models will be deployed, trained, and used for data extraction. This framework has different stages that will be discussed in this research paper. They include Taxonomy, which is used to define document type and type of fields that need to be extracted; Digitization, which converts text from documents into machine readable form; and Document Classification, which classifies the type of document so that specific fields can be retrieved; Document Classification Validation that is used to classify, using human intervention, and correct the data fetched if needed, Classification Training that is used to pass human validated information to classifiers for improving future predictions, Data Extraction that is used to capture the information required for identified document type, Data Extraction Validation that is used for reviewing and correcting classification results, Data Extraction Training that is used to pass human validated extracted results back to the extractors for improving future predictions, and Data Consumption. In this paper, each stage of the document understanding framework is discussed. [2]

There's also a template for the document understanding process in the studio. It is a fully functional UiPath studio project template based on a document processing flowchart. It lets any developer start a small-scale scale implementation or a large-scale enterprise project using this project. This architecture is typical for both attended and unattended implementations. This template can be found in the studio templates tab. This template has documentation in the project folder to access. It has preconfigured document types in taxonomy and a classifier

to differentiate between these classes and extractors [3]. Like Re-Framework, this can be used as a framework that can be adapted to the needs. The Document Understanding framework can be found in the UiPath.IntelligentOCR.Activities package. Once this package is installed, the Taxonomy Manager wizard appears in the UiPath Studio's design ribbon. This package contains all the core activities related to document understanding. Document understanding ML models run in the AI Center, the required infrastructure.

II. Document Understanding Framework Components:

This research paper discusses different components of the document understanding framework. Different components such as taxonomy, digitization, document classification, document classification validation, document classification training, data extraction, data extraction validation, data extraction training, and data consumption. This paper mentions how each part of the framework is helpful. This research study also discusses using ML skills in UiPath and customized ML algorithms for automating unstructured data extraction. While semi-structured data can be retrieved using document understanding, unstructured data with high variability can be retrieved using ML skill, which is part of the UiPath AI center.

III. Taxonomy:

The taxonomy contains the metadata and is a collection of document types. Document type is a type of document that is used by business processes. Some examples of those types are forms, contracts, invoices, medical records, etc. Each document type has a specific set of fields. The field is a type of information in the document, such as date, invoice number, amount, etc. Taxonomy will help in document classification by classifying incoming documents into certain types so that data can be retrieved for those document types. A hierarchical structure can be created containing group: different categories can be grouped; category: different document types can be created under a category; and document type: can be created as a group or part of a single document. For example, the group can be entered by price name, and the category can be IT, HR, Finance, and document type can be invoices, receipts, resumes, etc. Taxonomy contains a list of fields that allow the configuration of various extraction methods and rules based on the data schema: the structure of document type. The taxonomy structure is stored in a taxonomy.json file, which is unique for a project and can be accessed from the document processing folder within the UiPath Studio project. This file is automatically generated when Taxonomy Manager, which is used to load taxononomy.json, is created. Once the structure of documents is created, the next step is Load Taxonomy. The output of this activity is a document taxonomy, in which all the details related to taxonomy.json are stored. As mentioned in Figure 1- Structure of Documents in Taxonomy Manager, documents can be grouped based on the department (Revenue Cycle in this case), Category (Finance in this case), and Type (Invoice and MedicalRecords) [4]

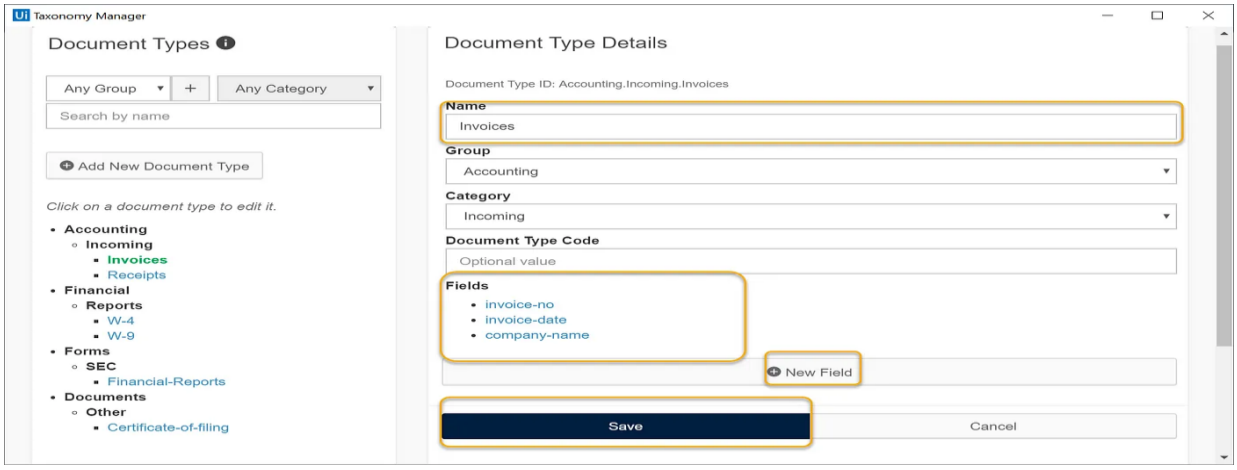

Figure 1- Structure of Documents in Taxonomy Manager

IV. Digitization:

In this step, the document information is converted into machine-readable text so that a robot can understand it. This first step will be applied to files processed through document understanding. This can be done for both digital and non-digital documents. The bot uses OCR (Optical Character Recognition) to read and extract data for non-digital documents. Use the activity called Digitize Document for this step. The input for this document is the Document file path of type string, output document text in string format, and document object model variable of type document, which has basic information such as name, content type, length of text, number of pages, and coordinates for each word that is identified in the file. Digitization is performed using the Digitize

Document activity. Though OCR can be applied, often, files that are processed are native PDF files, which aren't scanned, and hence, OCR is not required for such types of files. Once the OCR engine is configured, the bot executes OCR on files that are images. Some supported document types are .png, .gif, .jpe, .jpg, .jpeg, .tiff, .tif, .bmp. If there are multi-page images or TIFF files, OCR is applied to each page. Also, OCR is applied on PDF documents with no machine-readable content, also known as nonnative PDFs, containing images over a significant portion of the screen. As each use case is unique, different OCR engines must be tested before deciding what OCR engine would be best suited for the current use case. The settings for Digitize Document are as follows: ApplyOcrOnPdf – Determines whether OCR should be applied. Selecting Yes applies OCR, and selecting No doesn't apply OCR. The default value is Auto, letting the OCR algorithm decide whether OCR application is needed to be applied for the document; DegreeofParalelism-Activity tries to process as many pages in parallel as the value of numberofcores-1; and DetectCheckboxes- Detects available checkboxes and is set to true by default. For example, if the system has four core processors, and the degree of parallelism is set to -1, then three core processors are used, and hence three documents are processed simultaneously. In addition to extracting text from a PDF file, the Digitize Document activity adds pre-processing and post-processing methods to complex documents. You can combine this exercise with others that focus on document understanding.

Along with the Digitize Document activity, the OCR engine needs to be used when native content is not available. Images must have a minimum resolution of 50x50 MP and a maximum resolution of 9000x9000MP. [5] Every OCR engine among UiPath Document OCR, OmniPage OCR, Google Cloud Vision OCR, Microsoft Azure Computer Vision OCR, Microsoft OCR, Tesseract OCR, and Abby Document OCR, except Microsoft OCR, reports confidence. Abby OCR engines are not available by default in Studio.

V. Document Classification:

This step in document processing identifies the file type and classifies the file currently processed. If a file contains a single logical document type, such as an invoice or a medical record, then the output should have a single classification result. However, suppose the file has multiple types of documents (ex, page 1-5 invoice, page 6-10 invoice). In that case, the classification output will result in multiple classification results, each corresponding to the type of file based on page numbers in the input file. Only document types that are defined in the Taxonomy are attempted for classification. Document classification isn't required if the file type is only one. However, classification is required if multiple documents are in the input file. [6]

Classify Document Scope provides classifiers with classifying algorithms required to run, accepts multiple classifiers, allows document type filtering and confidence threshold settings, and reports classification information. Using Configure Classifiers provides customization of what document types are accepted by the classifier and a minimum threshold of acceptable confidence. The order in which classifiers are defined is also essential, and classifiers are executed with priority from left to right. A classifier's result will be accepted only if it reports acceptable document types and has a confidence threshold above the minimum level set for that classifier. Different classifiers are Keyword Based Classifier, Intelligent Keyword Classifier, FlexiCapture Classifier, and Machine Learning Classifier. In addition, we can build any classifier by using public document processing contracts, allowing us to implement customized classifiers for each use case. Keyword-based classifiers search for repeated patterns of words in a document to classify them. The algorithm is built around the document titles and is based on the premise that document titles have a low degree of variability. The keywords usually appear in a document are configured for this classifier, and the corresponding document type is pointed. This algorithm can work with single-string entries or multiple-string entries. This classifier is ideal if files contain one and only one type of document. The Intelligent keyword classifier uses a word vector to match in a file for document classification. It is based on the premise that each document type has specific common keywords. This algorithm finds the closest word vector a file is similar to and reports the highest-scoring document type with matched words. This algorithm also has splitting capabilities, meaning it can report more than one type of file based on the page range for each type. Hence, this algorithm is advantageous if a document has multiple file types. Flexicapture classifier classifies flexicapture definition files or a flexicapture classifier file, which can be found in UiPath.Abbyy.Activities package. This can only be used within the Classify Document Scope activity. The machine learning classifier uses a machine learning algorithm, which is defined as an ML skill in the UiPath AI center, to perform document classification tasks.

VI. Document Classification Validation:

This is an option step but a recommended one that sets up a human review for validation. It enables subject matter experts to review and correct the classified results if necessary. Correct classification results mean that the next steps in the Document Processing Framework are performed on the proper classification. [7] This is especially helpful when dealing with multiple document types within a single file. If the classification is wrong, if too few or too many pages are classified within a class, data passed downstream for the data extraction component would be applied to the wrong content. As 100% accuracy is needed for this reason, there are no other

ways of validating that the classified information is correct. Humans can validate results through the Classification of Station. This can be configured as an attended activity using Present Classification Activity or as an unattended action center task using Create Document Classification Action and Wait for Document Classification Action and Resume activities. A classification station can be used as a stand-alone tool with an action center. The file must be available on the user's machine to use this activity in attended automation. This activity can also be configured in unattended mode using the orchestration process of ReFramework and Action Center tasks. This approach increases productivity as it doesn't need to wait for the current transaction until it's finished. Also, the dependency on making files available on the user's machine can be eliminated.

VII. Document Classification Training:

This activity helps give feedback to the classifiers and lets them learn from human input. However, we must remember that not all Machine learning algorithms are retrainable. Some algorithms can't be retrained as there will be no retraining option. The project can be built without a training component, which is not mandatory. However, it is recommended that training be configured as classifiers can gather their training data and perform updates by taking human-ingested data. They'd become self-learning algorithms that can teach them how to act better in the future based on the corrected data by humans. Classifier training is done through the scope of Train Classifiers. As training should be done after document classification, only human-validated data should be sent to classifiers to ensure accuracy in training. Classification training is recommended in both success and failure cases, as both instances help machine learning algorithms learn. Keyword-Based Classifier Trainer is a training activity for the Keyword-Based Classifier Intelligent, and Keyword Classifier Trainer is a training activity for the Intelligent Keyword Classifier. [8]

VIII. Data Extraction:

Data extraction helps identify specific information from documents for extraction. The information that has to be retrieved from Taxonomy- and any field not previously defined in the Taxonomy cannot be retrieved during data extraction. Suppose there are multiple types of documents in the same file. In that case, classification will be run for each type of document separately, and the data extraction step must be run for every classification type. This is done through Data Extraction Scope activity. Multiple extractors can be used in this activity to extract various fields. For example, specific fields can be reliably extracted by certain extractors. It is recommended that different extractors for the fields in question be tested and that a combination of extractors is used if required. This activity provides all extraction algorithms necessary algorithms required for inputs, allows more than one extractor, provides field level mapping, taxonomy mapping, and minimum threshold setting, and reports extraction results in a unified manner. It is possible to choose which fields are extracted by which extractor. Even fallback rules can be applied: if an extractor doesn't extract the results with the required confidence level, call up the backup extractor. The extractors configured will be executed from left to right. If the Data Extraction Scope activity doesn't request any field from a given extractor, then that extractor isn't executed. The available Extractors are Regex Extractors, Form Extractors, Intelligent Form Extractors, Machine Learning Extractors, and FlexiCapture Extractors. The regex-based extractor is suitable for predictable cases. For instance, once a regex is configured, a Regex Extractor would be a good choice if the information is always available in the document. This extractor is unsuitable if a document has a high degree of variability. Also, the Regex extractor doesn't have learning capabilities and can be used in the Classify Document scope to classify the incoming document based on the keywords configured. Form extractor is used if some fields need to be extracted based on position inside the document, and it relies on templates defined at the design stage. The activity supports simple and table field extraction and can detect a signature field. If there are many different layouts, it is recommended to look into different extractors. This extractor allows the definition of multiple templates for the same type of document and identifies the best matching template for the incoming document. Intelligent Form Extractor is similar to form extractor but is enhanced, allowing handwriting and signature detection. This may be helpful in cases where a check needs to happen, whether the document may be printed or handwritten, or whether the form is signed. Machine Learning Extractor is the means to consume UiPath Document Understanding Models and is based on ML algorithms. The ML approach is highly suggested in the case of structured or semi-structured documents whose layouts vary significantly. Using the ML models, this extractor learns and infers values from extracted fields, even from the documents and layouts the algorithm never encountered. There are several applications for the machine learning model: either with custom-trained machine learning models beginning from the UiPath Document Understanding accessible models or with one of UiPath's public Document Understanding endpoints if you want to employ generic models targeting certain document types. To use this extractor, UiPath's public Document Understanding endpoints for data extraction need to be used, or machine learning models hosted in the AI Center need to be used. Set up the ML Skill argument of the activity with the appropriate choice from the AI Center-hosted ML skills list if you are utilizing the Machine Learning Extractor with a deployed ML skill. If you

would like the extraction process to be based on a collection of FlexiCapture definition files, use the FlexiCapture Extractor, which is available in the package UiPath.Abbyy.Activities. [9]

In the ML extractor, a list of fields will be available, and each field needs to be selected based on the requirement. Figure 2- List of available fields for ML extractor. Sometimes, an extractor could reliably extract a few fields, but another extractor works better for another set of fields. In this case, it is recommended to perform trial and error to decide what extractor must be used for what field. By default, a drop-down of tags will be available, and they need to be selected. Training activities need human intervention; using them once the Bot is deployed to Prod is not recommended. However, the Present Validation Station is highly recommended during development as training can be provided, and extracted data can be validated before deployment to Prod. In the digitized document activity, the degree of parallelism Can be configured. This is based on how many files can be processed simultaneously. If set to the default value of -1 and the number of cores is 3 for the processor, 3-1=2 documents will be processed in parallel. Exported results are stored in a variable called DataSet. For each row in this data set, values of the fields and corresponding confidence scores can be captured into a document. If we use the Present Validation Station after extracting results, the confidence score of every field will be 1, as humans would already have validated the fields. Hence, the Present Validation Station after Data Classification is used effectively. Confidence is not a member of extraction results and must be retrieved before use. For each item in extractionResults.ResultsDocument.Fields(0) extract the value of the item.Values(0).Confidence. The type argument for each loop is UiPath.DocumentProcessing.Contracts.Results.ResultsDataPoint. Based on this confidence score in the production run, the action center task of creating the task can be generated for manual review. In this case, the bot doesn't wait for the task to finish and goes to the next transaction, optimizing the human-in-the-loop process.

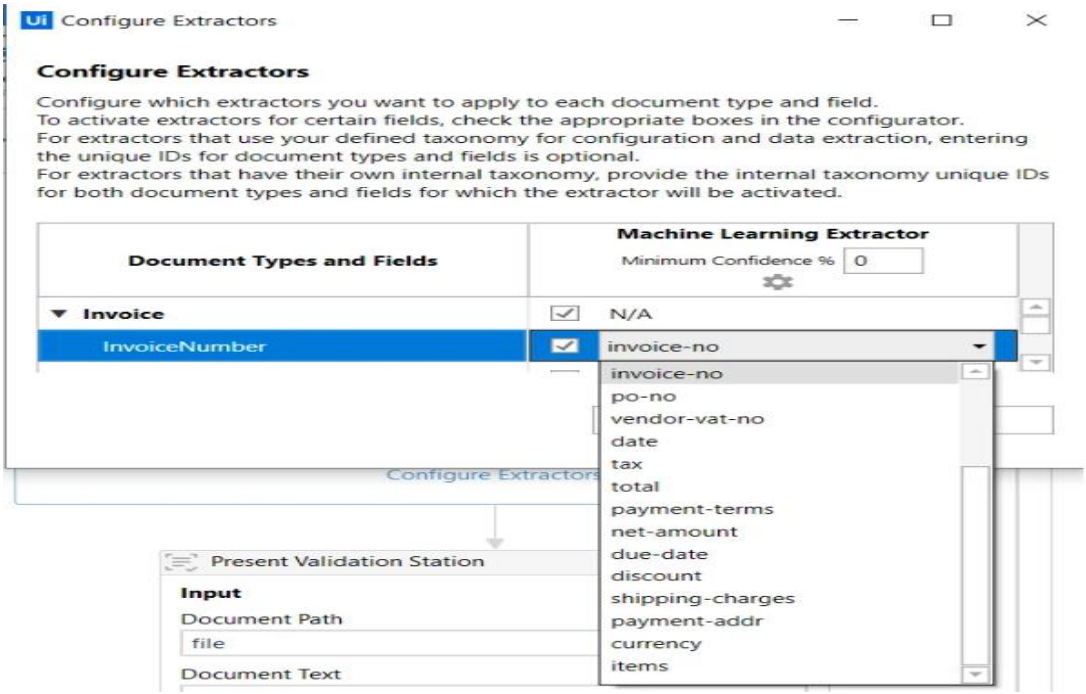

Figure 2- *List of available fields for ML extractor*

IX. Deploying Out-of-the-Box and Customized Machine Learning Models:

UiPath also allows customized machine-learning models to be used. Using ML packages involves these steps: collecting sample data and requirements of data points to be retrieved; labeling documents using data manager; downloading labeled documents as training dataset; downloading labeled documents as evaluation dataset; running training pipeline; evaluating the performance of ML model; deploying the ML model as ML skill in AI center; and Selecting the ML skill in the xaml workflow. These out-of-the-box ML algorithms classify and extract commonly occurring data points from semi-structured or unstructured documents. There's no option to define a template; hence, it is the template-less approach. [10] Document Understanding contains multiple ML Packages split into five main categories: UiPathDocumentOCR , which is a nontrainable ML model that can be used with the UiPath Document OCR engine as part of Digitize Document activity; UiPathDocumentOCR_CPU that can be used just like UiPathDocumentOCR ML Package with the specific improvements in speed, reduced accuracy, and optimized CPU; DocumentUnderstanding, which is retrainable model to extract common data points and training is a must for this ML package; Document Classifier; and Out-of-the-box Pre-trained ML

Packages, which can be customized to extract additional fields or support additional languages. Users can upload structured or unstructured datasets to train Machine Learning models. These data sets may need to be labeled before using for training. By using the endpoint and URL, the data set can be made public. Different options on the AI Center page are Datasets, Data Labelling, ML packages, Pipelines, ML skills, ML Logs, and Settings. The recommendation is to have two datasets: one for training and the other for evaluation purposes. Usually, 80% of data points are set aside for training, and 20% are used for evaluation. [11]

There are three types of pipelines, and document understanding ML models can run all of them: Training pipelines, evaluation pipelines, and Full Pipelines. Once run, the pipeline has associated outputs and logs. While training pipelines can fine-tune ML models with data from validation station, full pipelines can auto-finetune an ML model. The training pipeline is run on a test data set, and the training pipeline is run on evaluation data. Full pipeline allows configuration of both input dataset and evaluation dataset. The first step to deploying an ML model in the AI center is to create a project after entering the project name. The next step is to create and upload a data set. Next, a data labeling session is configured, which takes the new dataset or existing data set as input for labeling. The next step is creating an ML package name and an ML skill. It is recommended for production that GPUs be available for runs as the data is vast and needs to be processed using OCR. However, the processing can happen with just the CPUs, but the processing times may increase significantly compared to processing times using GPUs – a faster process. Once the ML skill is available after deployment, click on modify current deployment and copy the URL, which is the endpoint of UiPathDocumentOCR for later use. An out-of-the-box document understanding ML package can be deployed in the AI center. Click on ML Packages, upload a zip file, choose JSON type input and Python version, and click Create. Then, go to ML skills and create a new ML skill, which could take up to 30 minutes to reflect in the User Interface. The successful deployment of the ML model in the AI center has been completed. Once the ML Skill is created, double-click the ML Skill and go to Modify current deployment; switch the toggle to make the ML Skill public, and Copy the URL to use the created ML skill in the studio.

X. Conclusion:

In conclusion, Document processing using RPA is ridden with challenges. The data extraction can be configured easily if the document has a slight variation configured with regular expressions. However, if the structure varies, that makes the document semi-structured or unstructured, or if the document is a scanned image containing handwritten information, extracting information using traditional means such as regular expressions would be unreliable. In these cases, using a document understanding framework in UiPath would be a game-changing solution to extract data. As mentioned in this research paper, the data from semi-structured or unstructured documents can be extracted reliably using the document understanding framework and the steps in this framework. The built-in Machine Learning algorithms can be used for many types of documents that are part of business processes. However, in some cases, such as sentiment analysis, customized ML algorithms can be configured using out-of-the-box ML packages and can be used after defining ML skill. Creating Training pipelines, evaluation pipelines, and Full Pipelines would complete the process of defining customized ML algorithms. As mentioned in this paper, each field can be extracted with a certain confidence, and based on the confidence threshold, the transaction can be sent to the AI center as an action center task. This way, fields that aren't retrieved with as much confidence can be sent for human review, while the bot can process the fields that were retrieved with reasonable confidence.

References

[1] T. Venugopal, "Automating Invoice Processing in Fund Management: Insights from RPA and Data Integration Techniques," *Journal of Computational Analysis and Applications,* vol. 28, 2020.

[2] T. e. al., "Invoice Processing Automation," 2021.

[3] [Online]. Available: https://docs.uipath.com/document-understanding/standalone/2022.4/user-guide/taxonomy-overview. [Accessed August 2023].

[4] [Online]. Available: https://medium.com/@msharma697/uipath-document-understanding-decoding-taxonomy-8a205ec0d2e6. [Accessed August 2023].

[5] [Online]. Available: https://docs.uipath.com/document-understanding/standalone/2022.4/user-guide/digitization-ocr-engines. [Accessed August 2023].

[6] [Online]. Available: https://docs.uipath.com/document-understanding/standalone/2022.4/user-guide/document-classification-overview. [Accessed September 2023].

[7] [Online]. Available: https://docs.uipath.com/document-understanding/standalone/2022.4/user-guide/document-classification-validation-overview. [Accessed September 2023].

[8] [Online]. Available: https://docs.uipath.com/document-understanding/standalone/2022.4/user-guide/document-classification-training-overview. [Accessed September 2023].

[9] [Online]. Available: https://docs.uipath.com/document-understanding/standalone/2022.4/user-guide/data-extraction-overview. [Accessed September 2023].

[10] [Online]. Available: https://docs.uipath.com/document-understanding/standalone/2022.4/user-guide/about-ml-packages. [Accessed September 2023].

[11] [Online]. Available: https://docs.uipath.com/document-understanding/standalone/2022.4/user-guide/du-in-as-deploy-an-out-of-the-box-ml-package. [Accessed September 2023].